www.icgst.com

Enhancing the Minimum Average Scheduling Algorithm (MASA) based on Makespan Minimizing

Afaf Abd Elkader
Dept. of mathematics, Computer science Division, Faculty of science, Al-Azhar University
Cairo, Egypt
afaf2azhar@azhar.edu.eg

Abstract
The process of assigning tasks to resources with the aim of optimizing some objective is known as scheduling. Many algorithms are used to schedule tasks on their resources. One of these algorithms is the MASA scheduling algorithm which depends on max-min scheduling algorithm. A drawback of the max-min scheduling algorithm is that; the execution of tasks with maximum execution time first, increases the makespan, and leads to a delay in executing tasks with minimum execution times. The proposed algorithm e-MASA selects the task whose completion time is equal to (or the nearest to) the arithmetic mean of the minimum completion times of the remaining tasks. The proposed algorithm minimizes the makespan.

***Keywords**: Distributed system Scheduling, Scheduling algorithm, Max – min algorithm, Minimum Average Scheduling Algorithm, e-MASA*

Nomenclature

M	Number of Resources
N	Number of Tasks
MASA	Minimum Average Scheduling Algorithm
T_i	Task
R_j	Resource
r_j	Ready Time of Resource R_j
C_{ij}	Completion Time for Task T_i on Resource R_j
E_{ij}	Execution Time for Task T_i on Resource R_j
U	Set of all Tasks
OLB	Opportunistic Load Balancing
MET	Minimum Execution Time
MCT	Minimum Completion Time

1. Introduction
Scheduling is an algorithm design field which has a great attention. In a scheduling problem there are a set of resources $R=\{R_1,R_2,\dots, R_m\}$ and a set of tasks $T=\{T_1,T_2,\dots, T_n\}$. Each task is assigned to a resource in order to maximize or to minimize an objective function. Scheduling can be classified as either static or dynamic. In static scheduling, the information regarding all the resources as well as the tasks is assumed to be known in advance and each task is assigned once to a certain resource. In dynamic scheduling, the task allocation is done while the task executes where it is not possible to find execution times. Several algorithms for scheduling tasks have been developed to improve efficiency [1], [2]. The goal of the scheduling problem is to optimize an objective functions such as response time, makespan, CPU utilization, throughput, waiting time or turnaround time [3],[4].

- Response time: the amount of time from submission of a task to the first response
- Makespan: the maximum completion time
- CPU utilization: keeping the CPU as busy as possible
- Throughput: the number of processes that are completed per time unit
- Waiting time: the amount of time that the task is waiting in the ready queue
- Turnaround: the time between starting and finishing

This paper aims to introduce an efficient algorithm for scheduling tasks on resources with minimizing the makespan $C_{max}=max_i\{C_i\}$.

The rest of the paper is ordered as follows: Section (2) focuses on some related work Section (3) focuses on MASA Scheduling Algorithm Section (4) presents the proposed algorithm e-MASA Section (5) describes Experimental data Section (6) concludes the paper and presents future work.

2. Related work
Many algorithms have been proposed to schedule tasks on resources:

OLB (Opportunistic Load Balancing)
This algorithm assigns each task, in arbitrary order, to the next resource that is expected to be available, regardless of the task's expected execution time on that resource [2]. The idea behind OLB is to keep all resources as busy as possible. One advantage of OLB is its simplicity. However, because OLB does not consider expected task

execution times, the scheduling it finds can result in a very poor makespan.

MET (Minimum Execution Time)

The Minimum Execution Time algorithm assigns each task, in arbitrary order, to the resource with the best expected execution time for that task, regardless of that resource's availability [2]. The motivation behind MET is to give each task to its best resource. This can cause a severe load imbalance across resources.

MCT (Minimum Completion Time)

The MCT algorithm assigns each task, in arbitrary order, to the resource with the minimum expected completion time for that task [2]. This causes some tasks to be assigned to resources that do not have the minimum execution time for them.

Min–Min algorithm:

The Min-min algorithm begins with the set U of all unscheduled tasks. Then, the set of minimum completion times, $\{min_j \{C_{ij}\}\}$ for each task T_i in the set U, is found. From this set of minimum completion times, the task with the overall minimum completion time is selected and assigned to the corresponding resource. Last, the newly scheduled task is removed from the set U, and the process repeats until all tasks are scheduled (i.e., U is empty) [2]. Min-min is based on the minimum completion time, as is MCT. However, Min-min considers all unscheduled tasks during each scheduling decision and MCT only considers one task at a time [5].

Max–Min algorithm:

The Max-min algorithm is very similar to Min-min. The Max-min also begins with the set U of all unscheduled tasks. Then, the set of minimum completion times, $\{min_j \{C_{ij}\}\}$ for each task T_i in the set U, is found. From this set of minimum completion times, the task with the overall maximum completion time is selected and assigned to the corresponding resource. Last, the newly scheduled task is removed from U, and the process repeats until all tasks are scheduled (i.e., U is empty) [5].

Duplex algorithm:

The Duplex algorithm is a combination of the Min-min and Max-min algorithms. The Duplex performs both of the Min-min and Max-min algorithms and then uses the better solution [2].

RASA algorithm:

It uses the Max-Min, Min-Min algorithms. If the number of available resources is odd, the Min-min algorithm is applied to assign the first task. On the other hand, if the number of available resources is even, the Max-min algorithm is applied. The remaining tasks are assigned to their appropriate resources by one of the two algorithms alternatively. If the Min-min algorithm is applied to assign the first task to resource, the next task will be assigned by the Max-min algorithm. In the next round the task assignment begins with an algorithm different from the last round and so on [6].

Improved Max-Min Algorithm:

This algorithm uses the advantages of Max-Min and solves its drawbacks. It is concerned with the number of the resources and the tasks. It is based on the expected execution time instead of completion time. The algorithm calculates the expected completion time of the tasks on each resource. Then the task with the maximum expected execution time (Largest Task) is assigned to the corresponding resource that produces the minimum overall completion time (Slowest Resource). The scheduled task is removed from meta-tasks and all the corresponding times are updated. The traditional max-min algorithm is applied to the remaining tasks [7].

Enhanced Max-min Task Scheduling Algorithm:

This algorithm introduced a unique modification of Improved Max-min task scheduling algorithm [8]. It assigns the task with average execution time (average or nearest greater than average Task) to the slowest resource which produces minimum completion time.

MASA (Minimum Average Scheduling Algorithm):

The MASA selects the $\lfloor m/7 \rfloor$ (floor the number of resources divided by 7)tasks that have (minimum average execution times or nearest greater than minimum average execution times). These tasks are then assigned to the corresponding resources and the traditional Max-min algorithm is then applied [9]. This algorithm is described below.

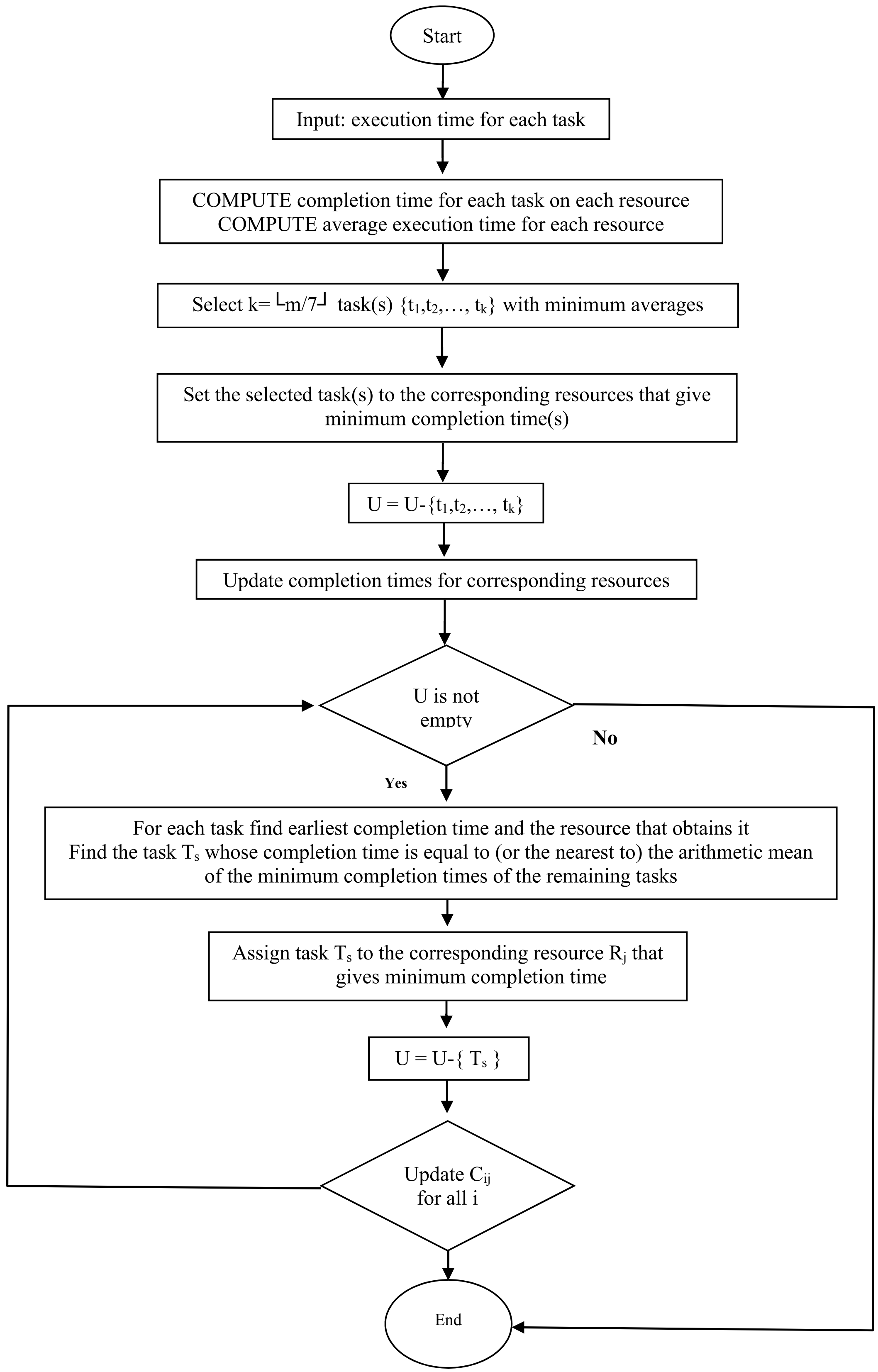
3. MASA (Minimum Average Scheduling Algorithm)

The MASA algorithm improves the Enhanced Max-min algorithm. Instead of selecting average or nearest greater than an average task, the MASA selects $\lfloor m/7 \rfloor$ tasks that have (minimum average execution times or nearest greater than minimum average execution times). The selected tasks are then assigned to the corresponding resources and the traditional Max-min algorithm is then applied. This algorithm decreases the makespan because it is does not depend on the largest average but assigns minimum average tasks to faster resources which produce minimum completion times.

4. The proposed algorithm: e-MASA (Enhanced Minimum Average Scheduling Algorithm)

The MASA algorithm depends on Max-min algorithm. A drawback of the Max-min algorithm is that, selecting tasks with maximum execution times first, increases the makespan and leads to a delay in executing tasks with minimum execution times. The proposed algorithm e-MASA selects the task whose completion time is equal to (or the nearest to) the arithmetic mean of the minimum completion times of the remaining tasks. The proposed algorithm gives a minimum makespan compared to the MASA algorithm.

e-MASA (Enhanced Minimum Average Scheduling Algorithm) Flowchart



e-MASA (Enhanced Minimum Average Scheduling Algorithm):

1) Input execution time for each task
2) For all tasks t_i in U
3) For all resources R_j
4) $C_{ij} \leftarrow E_{ij}+r_j$
5) For all resources R_j, Compute average execution times
6) Select k= $\lfloor m/7 \rfloor$ task(s); $\{t_1,t_2,\ldots, t_k\}$ with minimum averages
7) Set the selected task(s) to the corresponding Resource (s) R_j that gives minimum completion time
8) U $\leftarrow$U-$\{t_1,t_2,\ldots, t_k\}$
9) For all corresponding resources j and all selected task (s) T_i, Set $C_{ij} \leftarrow C_{ij}+ E_{ij}$
10) While there are tasks in U
11) For each task find earliest completion time and the resource that obtains it
12) $T_s \leftarrow$the task whose completion time is equal to (or the nearest to) the arithmetic mean of the minimum completion times of the remaining tasks
13) Assign task T_s to the corresponding resource R_j that gives minimum completion time
14) U$\leftarrow$U-$\{ T_s \}$
15) For all i, Set $C_{ij} \leftarrow C_{ij}+ E_{sj}$
16) End while

The proposed algorithm has complexity O(mn^2) as that of the MASA, where m is the number of resources in the system and n is the number of tasks.

A Comparison between the e-MASA and MASA algorithms

Given a set of tasks, U with resources and execution times shown below in Table 1.

Table1. Execution times of tasks

Resources / Tasks	R0	R1	R2	R3	R4	R5	R6
T0	2	8	5	1	10	5	9
T1	9	3	5	6	6	2	8
T2	2	2	6	3	8	7	2
T3	5	3	4	3	3	2	7
T4	9	6	8	7	2	9	10
T5	3	8	10	6	5	4	2
T6	3	4	4	5	2	2	4
T7	9	8	5	3	8	8	10
T8	4	2	10	9	7	6	1
T9	3	9	7	1	3	5	9
T10	7	6	1	10	1	1	7
T11	2	4	9	10	4	5	5
T12	7	1	7	7	2	9	5
T13	10	7	4	8	9	9	3
T14	10	2	4	6	10	9	5
T15	1	8	7	4	7	2	6
T16	5	3	1	10	8	4	8
T17	3	7	1	2	7	6	8
T18	6	5	2	3	1	1	2
T19	5	7	1	8	2	8	8

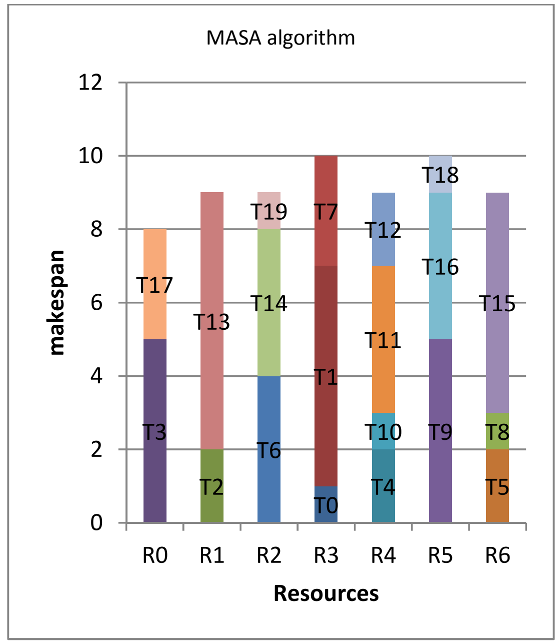


Figure 1: (a) Gantt chart of MASA algorithm

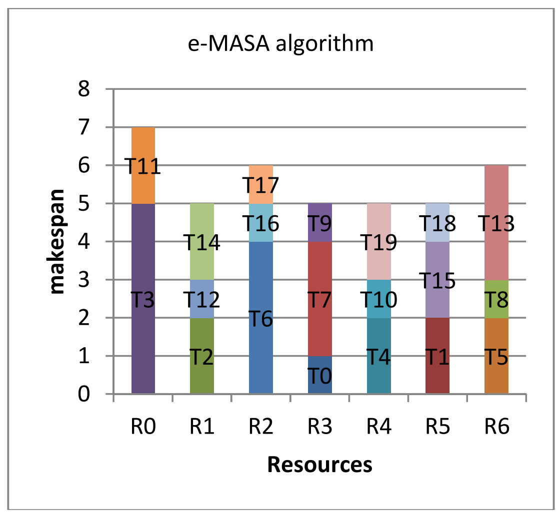


Figure 1: (b) Gantt Chart of e-MASA Algorithm

Figure 1 describes the makespan using both the MASA and the e-MASA algorithms. The MASA algorithm produces a makespan=10 ms while e-MASA produces a makespan=7 ms. The results show that e-MASA produces a makespan smaller than that of the e-MASA algorithm.

5. Experimental Data

A simulation is made for the MASA algorithm and the e-MASA algorithm to analyze their performance. The simulation is written using the C++ language. The model consists of m resources, n tasks with m varying from 200 to 1000 and n varying from 1000 to 5000. First, the number of resources is fixed at m=100 and the makespan is calculated for different values of the number of tasks n: 1000 until 10000 tasks. The values of execution times

of tasks are chosen randomly. As the number of tasks increases, the makespan increases with e-MASA giving smaller results. Figure 2 plots the makespan versus the number of tasks with m=100.

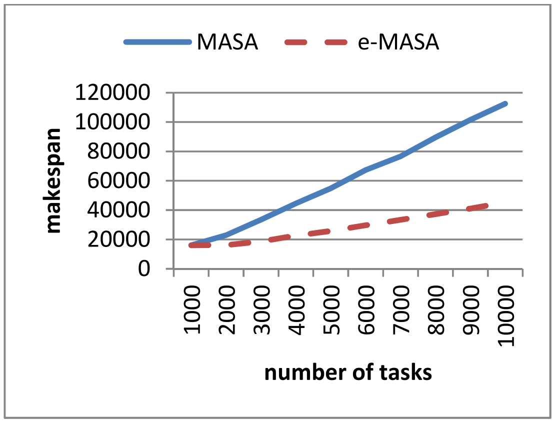


Figure 2:The makespan versus the number of tasks for both MASA and e-MASA algorithms

Then, the number of tasks is fixed at n=10000 and the makespan is calculated for different values of the number of resources m: 200 to 1000 resource. The execution times are chosen randomly from 2000 to 5000. As the number of resources increases, the makespan decreases with e-MASA giving smaller results than the MASA algorithm. Figure 3 plots the makespan versus the number of resources with n=10000.

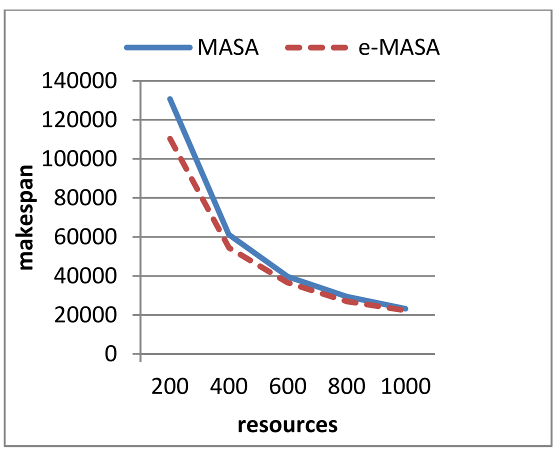


Figure 3: The makespan versus the number of resources for both MASA and e-MASA algorithms

6. Conclusions

The MASA algorithm depends on max-min scheduling algorithm which executes tasks with maximum execution times first. This increases the makespan, and leads to a delay in executing minimum execution times tasks. The proposed algorithm e-MASA selects the task whose completion time is equal to (or the nearest to) the arithmetic mean of the minimum completion times of the remaining tasks. Comparing the makespan produced by the two algorithms; the results show that the e-MASA algorithm improves the makespan.

Future Work

The proposed algorithm may be applied in parallel computing, cloud computing, distributed systems and operating systems.

7. References

[1] A. Chandak, B. Sahoo, A. Turuk, "An overview of task scheduling and performance metrics in grid computing", International Journal of Research and Reviews in Computer Science, Vol. 2(2), pp. 30–33,2011.

[2] Braun TD, Siegel HJ, Beck N, Boloni LL, Maheswaran M, Reuther AL, et al, "A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems", J Parallel Distributed Computing, Vol. 61(6), pp.810–837, 2001.

[3] Elzeki OM, Rashad MZ, Elsoud MA, "Overview of scheduling tasks in distributed Computing systems ", Int. J Soft Computing and Engineering, Vol. 2(3), pp.470–475, 2012.

[4] Neetu Goel, R.B. Garg, "A Comparative Study of CPU Scheduling Algorithms", International Journal of Graphics & Image Processing, Vol. 2(4), pp.245-251,2012.

[5] Pinal Salot, "A Survey of various scheduling algorithm in cloud computing environment", IJRET - International Journal of Research in Engineering and Technology, Vol. 2(2), pp.131-135, 2013.

[6] Saeed Parsa, Reza Entezari-Maleki, "RASA: A New Grid Task Scheduling Algorithm", International Journal of Digital Content Technology and its Applications, Vol. 3(4), pp. 91-99, 2009.

[7] O. M. Elzeki, M. Z. Reshad and M. A. Elsoud, "Improved Max-Min Algorithm in Cloud Computing", International Journal of Computer Applications, Vol. 50(12), pp.22-27, 2012.

[8] Upendra Bhoi, Purvi N. Ramanuj, "Enhanced Max-min Task Scheduling Algorithm in Cloud Computing", International Journal of Application or Innovation in Engineering & Management, Vol. 2(4), pp. 259-264, 2013.

[9] Kamal ElDahshan, Afaf Abd Elkader, Nermeen Ghazy, "Minimum Average Scheduling Algorithm, MASA, Performance Boosting Approach", Artificial Intelligence and Machine Learning Journal, Vol.16(1), pp. 23-29, 2016

Dr. Afaf Abd Elkader Abd Elhafiz is currently lecturer at Al-Azhar University. She is doing her research work in Scheduling algorithms.